CLAIMS

What is claimed is:

1. A method for replicating application program objects using semi-active or passive replication, wherein replicas of said objects are distributed across a plurality of networked computers for fault tolerance, said method comprising:

creating a primary replica of an object and one or more backup replicas of said object; and

the step of communicating message ordering information from said primary replica to said backup replicas for achieving consistent message ordering among said primary and backup replicas.

2. A method as recited in claim 1:

wherein said communication of message ordering information to said backup replicas can occur after said primary replica has started processing said messages, and either before or concurrently with the transmission by said primary replica of its next message; and

wherein said communication of message ordering information may be achieved without the transmission of additional messages beyond those required for operation of the networked computers without fault tolerance.

3. A method as recited in claim 1, wherein said primary replica processes messages selected from the set of messages consisting essentially of synchronous requests, synchronous replies, asynchronous requests, asynchronous replies and one-way messages.

4. A method as recited in claim 3, wherein said messages are selected from the set of messages consisting essentially of TCP messages, SCTP messages, UDP messages, and remote procedure calls.

5. A method as recited in claim 1:

wherein a primary server replica processes requests received from a primary client replica in an order determined by a criterion selected from a set of message scheduling criteria consisting essentially of sequence number order, reception order, priority, deadline, fairness, and availability of resources; and

wherein said primary server replica generates message ordering information corresponding to the order in which it processes messages.

6. A method as recited in claim 1, wherein said primary replica piggybacks message ordering information on its messages and multicasts those messages to source object replicas and destination object replicas.

7. A method as recited in claim 1:

wherein said primary replica piggybacks message ordering information on its messages and multicasts those messages to destination object replicas;

wherein said destination object has a primary replica; and

wherein said primary replica of said destination object piggybacks said message ordering information on its messages and multicasts those messages to source object replicas.

8. A method for replicating application program objects using semi-active or passive replication, wherein replicas of said objects are distributed across a plurality of networked computers for fault tolerance, said method comprising:

communicating message ordering information from said primary replica to said backup replicas for achieving consistent message ordering among said primary and backup replicas;

wherein said communication of message ordering information to said backup replicas can occur after said primary replica has started processing said messages, and either before or concurrently with the transmission by said primary replica of its next message; and

wherein said communication of message ordering information may be

achieved without the transmission of additional messages beyond those required for operation of the networked computers without fault tolerance.

9. A method as recited in claim 8, wherein said primary replica processes messages selected from the set of messages consisting essentially of synchronous requests, synchronous replies, asynchronous requests, asynchronous replies and one-way messages.

10. A method as recited in claim 9, wherein said messages are selected from the set of messages consisting essentially of TCP messages, SCTP messages, UDP messages, and remote procedure calls.

11. A method as recited in claim 8:

wherein a primary server replica processes requests received from a primary client replica in an order determined by a criterion selected from a set of message scheduling criteria consisting essentially of sequence number order, reception order, priority, deadline, fairness, and availability of resources; and

wherein said primary server replica generates message ordering information corresponding to the order in which it processes messages.

12. A method as recited in claim 8, wherein said primary replica piggybacks message ordering information on its messages and multicasts those messages to source object replicas and destination object replicas.

13. A method as recited in claim 8:

wherein said primary replica piggybacks message ordering information on its messages and multicasts those messages to destination object replicas;

wherein said destination object has a primary replica; and

wherein said primary replica of said destination object piggybacks said message ordering information on its messages and multicasts those messages to source object replicas.

14. A fault-tolerant computer system, wherein application program objects are replicated using semi-active or passive replication and distributed across a plurality of networked computers, said system comprising:

application program object replication software executable on one or more networked computers in said system; and

means associated with said replication software for communication of message ordering information from a primary replica to one or more backup replicas and for achieving consistent message ordering among said primary and backup replicas.

15. A system as recited in claim 14:

wherein said communication of message ordering information to said backup replicas, either directly or indirectly, can occur after said primary replica has started processing said messages, and either before or concurrently with the transmission by said primary replica of its next message; and

wherein said communication of message ordering information may be achieved without the transmission of additional messages beyond those required for operation of said application program objects without fault tolerance.

16. A computer system as recited in claim 14, wherein said primary replica processes messages selected from the set of messages consisting essentially of synchronous requests, synchronous replies, asynchronous requests, asynchronous replies and one-way messages.

17. A computer system as recited in claim 16, wherein said messages are selected from the set of messages consisting essentially of TCP messages, SCTP messages, UDP messages, and remote procedure calls.

18. A computer system as recited in claim 14:

wherein a primary server replica processes requests received from a primary client replica in an order determined by a criterion selected from the set of message

scheduling criteria consisting essentially of sequence number order, reception order, priority, deadline, fairness, and availability of resources; and

wherein said primary server replica generates message ordering information corresponding to the order in which it processes messages.

19. A computer system as recited in claim 14, wherein said primary replica piggybacks message ordering information on its messages and multicasts those messages to source object replicas and destination object replicas.

20. A computer system as recited in claim 14, wherein said primary replica piggybacks message ordering information on its messages and multicasts those messages to destination object replicas;

wherein said destination object has a primary replica;

and wherein said primary replica of said destination object piggybacks said message ordering information on its messages and multicasts those messages to source object replicas.

21. A fault-tolerant computer system, wherein application program objects are replicated using semi-active or passive replication and distributed across a plurality of networked computers, said system comprising:

application program object replication software executable on one or more networked computers in said system; and

means associated with said replication software for communication of message ordering information from a primary replica to one or more backup replicas and for achieving consistent message ordering among said primary and backup replicas;

wherein said communication of message ordering information to said backup replicas, either directly or indirectly, can occur after said primary replica has started processing said messages, and either before or concurrently with the transmission by said primary replica of its next message; and

wherein said communication of message ordering information may be

achieved without the transmission of additional messages beyond those required for operation of said application program objects without fault tolerance.

22. A computer system as recited in claim 21, wherein said primary replica processes messages selected from the set of messages consisting essentially of synchronous requests, synchronous replies, asynchronous requests, asynchronous replies and one-way messages.

23. A computer system as recited in claim 22, wherein said messages are selected from the set of messages consisting essentially of TCP messages, SCTP messages, UDP messages, and remote procedure calls.

24. A computer system as recited in claim 21:

wherein a primary server replica processes requests received from a primary client replica in an order determined by a criterion selected from the set of message scheduling criteria consisting essentially of sequence number order, reception order, priority, deadline, fairness, and availability of resources; and

wherein said primary server replica generates message ordering information corresponding to the order in which it processes messages.

25. A computer system as recited in claim 21, wherein said primary replica piggybacks message ordering information on its messages and multicasts those messages to source object replicas and destination object replicas.

26. A computer system as recited in claim 21, wherein said primary replica piggybacks message ordering information on its messages and multicasts those messages to destination object replicas;

wherein said destination object has a primary replica; and

wherein said primary replica of said destination object piggybacks said message ordering information on its messages and multicasts those messages to source object replicas.

27. A computer program executable on one or more computers in a system of networked computers wherein application program objects are replicated using semi-active or passive replication and distributed across a plurality of said networked computers for fault tolerance, said program comprising:

instructions for communication of message ordering information from a primary replica of an object to backup replicas of said object for achieving consistent message ordering among said primary and backup replicas

wherein said communication of message ordering information to said backup replicas, either directly or indirectly, can occur after said primary replica has started processing said messages, and either before or concurrently with the transmission by said primary replica of its next message; and

wherein said communication of message ordering information may be achieved without the transmission of additional messages beyond those required for operation of said application program objects without fault tolerance.

28. A computer program as recited in claim 27, wherein said primary replica processes messages selected from the set of messages consisting essentially of synchronous requests, synchronous replies, asynchronous requests, asynchronous replies and one-way messages.

29. A computer program as recited in claim 28, wherein said messages are selected from the set of messages consisting essentially of TCP messages, UDP messages, SCTP messages and remote procedure calls.

30. A computer program as recited in claim 27:

wherein a primary server replica processes requests received from a primary client replica in an order determined by a criterion selected from the set of criteria consisting essentially of sequence number order, reception order, priority, deadline, fairness, and availability of resources; and

wherein said primary server replica generates message ordering information corresponding to the order in which it processes messages.

31. A computer program as recited in claim 27, wherein said primary replica piggybacks message ordering information on its messages and multicasts those messages to source object replicas and destination object replicas.

32. A computer program as recited in claim 27, wherein said primary replica piggybacks message ordering information on its messages and multicasts those messages to destination object replicas;

wherein said destination object has a primary replica;

and wherein said primary replica of said destination object piggybacks said message ordering information on its messages and multicasts those messages to source object replicas.

33. A computer program executable on one or more computers in a system of networked computers wherein application program objects are replicated using semi-active or passive replication and distributed across a plurality of said networked computers for fault tolerance, said program comprising:

a plurality of executable procedures configured to exploit multicast communication to achieve fault tolerance using semi-active or passive replication of objects with a primary replica and one or more backup replicas, and to maintain a consistent message delivery order across the replicas of an object to achieve strong replica consistency, despite asynchronous message reception, message loss and failure of the replicas, without requiring additional messages in fault-free conditions beyond those required in a system without replication.

34. A computer program as recited in claim 33, wherein wrapping and library interpositioning are used to divert messages from standard protocol stacks to said procedures so that application programs can be rendered fault-tolerant without modification to said application programs.

35. A computer program as recited in claim 34, wherein a primary replica of an object can receive and process a first message, and can multicast further messages as a result of processing said first message, prior to announcing an order

for said first message so that, in fault-free conditions, the end-to-end request/response time is not degraded by the need to transmit additional messages, or to wait for reception of additional messages, over and above the messages required in a system without replication.

36. A computer program as recited in claim 35, wherein for each pair of replicated objects between which messages are to be communicated, a multicast connection is established and every message multicast on said connection by a replica of a first object of said pair of objects is delivered to each replica of both objects, or to each replica of the second object of said pair of objects.

37. A computer program as recited in claim 36, wherein replicas of each object that participates in the multicast connection requires no knowledge of the number of replicas, or of the individual replicas, of the other object.

38. A computer program as recited in claim 37, wherein objects communicate by use of any or all of synchronous request messages, synchronous reply messages, asynchronous request messages, asynchronous reply messages or one-way messages.

39. A computer program as recited in claim 38, wherein an object can process messages in an order determined by priority, deadline, fairness, availability of resources or other message scheduling criteria not dependent solely on the order in which messages are generated, transmitted or received.

40. A computer program as recited in claim 39, wherein said primary replica of an object generates and multicasts announcements that report the order in which it has processed, or is processing, messages.

41. A computer program as recited in claim 40, wherein backup replicas of an object process messages in the order specified in said announcements that said primary replica generates and multicasts.

42. A computer program as recited in claim 41, wherein said primary replica of an object multicasts to said backup replicas of said object an announcement of the ordering of a message piggybacked on a synchronous request message, synchronous reply message, asynchronous request message, asynchronous reply message or one-way message, if such an application message is available, to avoid transmitting additional messages containing said announcement.

43. A computer program as recited in claim 42, wherein said primary replica of an object multicasts said announcements of the ordering of several messages in a single message.

44. A computer program as recited in claim 42, wherein said primary replica of an object multicasts to said backup replicas of said object said announcement of the ordering of a message piggybacked on the first synchronous request message, synchronous reply message, asynchronous request message, asynchronous reply message, one-way message or other message that said primary replica multicasts after it has processed, or while it is processing, said message.

45. A computer program as recited in claim 44, wherein said primary replica of an object multicasts said announcements of the ordering of several messages in a single message.

46. A computer program as recited in claim 45, wherein if no application message is available, said primary replica of an object multicasts said announcement of the ordering of a message in a null message.

47. A computer program as recited in claim 46, wherein said primary replica of an object multicasts said announcements of the ordering of several messages in a single message.

48. A computer program as recited in claim 41, wherein said primary replica or backup replica of an object can process a synchronous reply message without waiting to transmit or receive other messages and without generating, or

waiting for, an announcement of the order in which to process the reply message.

49. A computer program as recited in claim 48, wherein said primary replica of an object multicasts to said backup replicas of said object said announcement of the ordering of a message piggybacked on a synchronous request message, synchronous reply message, asynchronous request message, asynchronous reply message or one-way message, if such an application message is available, to avoid transmitting additional messages containing said announcement.

50. A computer program as recited in claim 49, wherein said primary replica of an object multicasts announcements of the ordering of several messages in a single message.

51. A computer program as recited in claim 49, wherein said primary replica of an object multicasts to said backup replicas of said object said announcement of the ordering of a message piggybacked on the first synchronous request message, synchronous reply message, asynchronous request message, asynchronous reply message, one-way message or other message that said primary replica multicasts after it has processed, or while it is processing, said message.

52. A computer program as recited in claim 51, wherein said primary replica of an object multicasts announcements of the ordering of several messages in a single message.

53. A computer program as recited in claim 52, wherein if no application message is available, said primary replica of an object multicasts an announcement of the ordering of a message in a null message.

54. A computer program as recited in claim 53, wherein said primary replica of an object multicasts said announcements of the ordering of several messages in a single message.

55. A computer program as recited in claim 41, wherein said primary replica of an object can process a synchronous request message, asynchronous request message, asynchronous reply message or one-way message before it generates and multicasts said announcement of the order in which to process that message, if such processing is consistent with sequence number order, reception order, priority, deadline, fairness, availability of resources or other message scheduling criteria.

56. A computer program as recited in claim 55, wherein said primary replica of an object multicasts to said backup replicas of said object said announcement of the ordering of a message piggybacked on a synchronous request message, synchronous reply message, asynchronous request message, asynchronous reply message or one-way message, if such an application message is available, to avoid transmitting additional messages containing said announcement.

57. A computer program as recited in claim 56, wherein said primary replica of an object multicasts announcements of the ordering of several messages in a single message.

58. A computer program as recited in claim 56, wherein said primary replica of an object multicasts to said backup replicas of the object said announcement of the ordering of a message piggybacked on the first synchronous request message, synchronous reply message, asynchronous request message, asynchronous reply message, one-way message or other message that said primary replica multicasts after it has processed, or while it is processing, said message.

59. A computer program as recited in claim 58, wherein said primary replica of an object multicasts announcements of the ordering of several messages in a single message.

60. A computer program as recited in claim 58, wherein if no application message is available, said primary replica of an object multicasts said

announcement of the ordering of a message in a null message.

61. A computer program as recited in claim 60, wherein said primary replica of an object multicasts said announcements of the ordering of several messages in a single message.

62. A computer program as recited in claim 41, wherein said primary replica of an object transmits a message containing the announcement of the ordering of a message over any multicast connection of which said primary replica is a participant.

63. A computer program as recited in claim 62, wherein said backup replicas of an object receive messages, multicast by said primary replica of said object, containing announcements of the ordering of messages, and in which said backup replicas of said object process those messages in the order specified by said announcements.

64. A computer program as recited in claim 63, wherein said primary replica of an object processes messages in the order specified in its announcements and said backup replicas of said object process messages in the order specified in the announcements that they received from said primary replica, so that said primary replica and said backup replicas process the same messages in the same order.

65. A computer program as recited in claim 64, wherein if the primary replica of a second object processes request messages from the primary replica of a first object, and if said primary replica of said second object multicasts a reply to the replicas of said first object, and if said primary replica of said second object fails so that a backup replica of said second object assumes the role of the primary, then said backup replica can obtain, from one of the other replicas of said second object or from one of the replicas of said first object, the messages and announcements of the ordering of those messages, so that said backup replica can process the same messages in the same order as said primary replica processed those messages before it failed.

66. A computer program as recited in claim 63, wherein the primary replica of a first object multicasts a message containing a method invocation of a second object, the primary replica of said second object includes an announcement of the ordering of said message in a message that it multicasts to the replicas of said first object, and said primary replica of said first object includes said announcement in a message that it multicasts to the replicas of said second object.

67. A computer program as recited in claim 66, wherein the backup replicas of said second object receive messages multicast by said primary replica of said first object, containing announcements of the ordering of messages, that the replicas of said first object received from said primary replica of said second object, which generated said announcements and piggybacked those announcements in messages that it multicast to the replicas of said first object.

68. A computer program as recited in claim 67, wherein the primary replica of an object processes messages in the order specified in its announcements and the backup replicas of said object process messages in the order specified in the announcements that they received from said primary, so that said primary replica and said backup replicas process the same messages in the same order.

69. A computer program as recited in claim 68, wherein if said primary replica of said second object processes request messages from said primary replica of said first object, and if said primary replica of said second object multicasts a reply to the replicas of said first object, and if said primary replica of said second object fails so that a backup replica of said second object assumes the role of the primary, then said backup replica can obtain, from one of the other replicas of said second object or from one of the replicas of said first object, the messages and announcements of the ordering of those messages, so that said backup replica can process the same messages in the same order as said primary replica processed those messages before it failed.